**Universidad de los Andes Ingeniería de Sistemas y Computación**

**Infraestructura Computacional**

**Trabajo Caso 1**

Presentado Por:

**Sebastián Salas 201225327**

**Nicolás Rozo 201225282**

El proyecto contiene cuatro clases que son Buffer, Mensaje, Cliente y Servidor. La clase buffer contiene la lista donde se almacena los mensajes, una lista de los threads activos, un contador de los clientes que no han terminado y el método main (inicializa toda la estructura con el número de clientes, el número de servidores y el número de consultas de cada uno de los clientes desde el archivo config.properties).

Primero, se inicializa una cantidad de clientes, que extienden de la clase thread, con un número de mensajes. Se corre el thread cliente ( *Cliente.run()* ) y este crea un mensaje y preguntara al buffer si tiene la capacidad de atender su mensaje ( *Buffer.encolarMensaje(…)* y *Buffer.agregarMensaje(…)* ). El cliente se dormira en el buffer en caso de no tener mas espacio. De lo contrario, el cliente registra el mensaje en el buffer y se duerme en el mensaje ( *Mensaje.dormirPadre()* ). En caso de que duerma, este solo se despertara en el momento en que un mensaje salga de la lista del buffer y este mismo notifica a todos los clientes dormidos en él para que compitan por la entrada ( *Buffer.remover(…)* ).

Luego de que se inicialicen los Clientes se inicializan los Servidores. Estos son threads que al correr ( *Servidor.run()* ) va estar esperando activamente a que un mensaje entre al buffer. Este conoce si la lista del buffer tiene mensajes o está vacía ( *Buffer.remover()* ). El servidor hace exclusión mutua de un mensaje ( *synchronized* ) para poder modificarlo y luego eliminarlo de la cola sin que otro servidor lo altere. Luego de responder el mensaje levanta al cliente que envió el mensaje ( *Mensaje.levantarPadre()*).

El cliente al ser atendido sale del buffer y si tiene otro mensaje vuelve hacer el proceso de mandarlo. En caso de que haya terminado, el cliente le notifica al buffer que ya acabo y el buffer reduce el conteo de cuantos clientes restantes ( *Buffer.termine()* ). En el caso en el que no haya clientes restantes, entonces los servidores paran su proceso ( *Servidor.detener()* ) y se acaba todo el programa.

El config properties es el archivo que administra y donde se modifica la cantidad de las variables (Limite del buffer, Cantidad de mensajes por Cliente, Cantidad de Clientes y Cantidad de Servidores) se encuentra en la carpeta Caso1->Resources.